IMPLEMENTING COLLABORATION MODERATOR SERVICE TO SUPPORT VARIOUS PHASES OF VIRTUAL ORGANISATIONS

J. A. Harding†* and R. Swarnkar‡
† Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University, Loughborough, LE11 3TU, UK
‡R&D, Accelrys Software, Inc, 334 Science Park, Cambridge, CB4 0WN, UK
* Corresponding author: j.a.harding@lboro.ac.uk

ABSTRACT

Research into Moderators, which support collaborative teams by proactively making team members aware of actions or potential problems which may affect them, began in the 1990s, in the context of supporting collaborations during concurrent engineering projects. This paper provides a background to the evolution of Moderators and explores their role in supporting virtual organisations. A Collaboration Moderator (CM) is an evolution of earlier moderators and is capable of behaving differently for different types of users and therefore caters for the varying requirements of individual users depending on the roles they have in the collaborations. This paper describes the architecture and components of a CM from an implementation perspective. Prototype CMs have been developed during the EU funded SYNERGY project, and two use cases for which the prototype CMs were implemented as services (a Pre-Creation use case and an Operational use case) are also discussed in this paper.

Keywords: Collaborations, virtual enterprises, virtual organisations, moderators, knowledge management support for collaborations.

1. INTRODUCTION

Virtual Organisations (VOs) are essentially comprised of a number of autonomous and semi-independent partners, each of which has a range of capabilities and resources at their disposal. These partners can be individuals, departments or whole organisations. These VOs are formed so that resources may be pooled and services combined with a view to exploiting a perceived market niche (Norman *et al.* 2004). Walton and Whicker (1996) describe the VOs as a series of cooperating "nodes" of core competence forming a value chain for addressing an opportunity. However, it can be argued that VOs are networks of competency nodes, rather than a series. Camarinha and Afsarmanesh (1999) define a VO as a temporary alliance of organisations that come together to share skills or core competencies and resources in order to better respond to business opportunities and whose cooperation is supported by computer networks.

The establishment of cooperation agreements and strategic partnerships between organisations is not a new phenomenon in the world of business. However the use of communication and information technologies to support virtualisation strategies has increased significantly with the advances in these areas. Organisations are entering into wider alliances and communities of practice through electronic means, and at the same time, internet technologies are adopted at both intra–firm and inter–firm levels to aid virtualisation strategies (Camarinha-Matos *et al.* 1998, Salazar *et al.* 2003). Virtualisation and working as cooperating teams bring many obvious advantages, however the interdependent nature of collaborative activities introduces complexities into the business processes. Indeed processes can be too complicated and time consuming, creating many obstacles to collaboration, and therefore the virtual organisation creation process is a critical to the successes of the network (Camarinha-Matos et al, 2009).

Knowledge sharing has been identified as one of the key factors governing the success of collaborations. However it has also been identified that effective sharing of knowledge and information is one of the major challenges facing virtual teams (Camarinha-Matos 2004). Ratcheva (2009) mentions that knowledge diversity in multidisciplinary teams can hinder their effective sharing of the knowledge. The difficulty of managing knowledge exchanges amongst team members can become a major barrier to any successful multidisciplinary activity. Lack of shared knowledge or awareness of team members' priorities can lead to sub–optimal decision processes, affecting the overall performance of the team. Therefore, tools and techniques for timely creation, sharing and reuse of appropriate knowledge are vital for collaborative team working.

This paper describes the real time operations of a collaboration moderator (CM). The CM exists to aid collaborations by raising awareness of the priorities and requirements of other contributors. Prototype CMs have been developed during the EU funded SYNERGY project (SYNERGY 2008) (Popplewell, 2008). The rest of this paper is organised as follows: the next section explains the earlier work leading to the development of the CM. Subsequent sections briefly describe an architecture relevant to the real time operations of the CM followed by implementation issues and case studies. The paper concludes with a section discussing the impact of such a tool on collaborative working and the feedback from users. A list of abbreviations is provided as an appendix as a quick reference for the reader.

2. Background

Research into Moderators began in the 1990s in the context of supporting collaborations during concurrent engineering (CE) projects. Information management was identified as the key to success in a CE environment (Harding and Popplewell, 1996). An individual partner in a CE project is likely to take decisions which unintentionally may cause disruptions in activities carried out by other partners. Therefore members of a CE team need to be aware of each other's activities, since lack of awareness can lead to situations like:

- Members not knowing that their actions have affected or can affect others
- Members not knowing that another's actions have affected or are going to affect them or others

The research into Moderators progressed through several research projects during the 1990s and 2000s and the concepts have been applied to different contexts, but throughout the research, a common aspect of their functionality has been to support collaborations and increase awareness between collaborating partners. Harding and Popplewell (1996) provided specifications of the Engineering Moderator to support teams working in a CE environment, by encouraging and facilitating communication and negotiations between design agents and other team members. Harding *et al.* (2003) developed a Manufacturing System Engineering Moderator to monitor design decisions and evaluate their significance to individual project team members and to communicate with the affected teams. Further details of the history and development of Moderators can be found in Harding *et al* (2007).

3. Collaboration Moderator

A collaboration moderator (CM) is a specialist application for supporting individual collaboration partners (and therefore the collaboration as a whole) by raising awareness of issues affecting the items of interest (IOIs) identified by the partners. A CM is an evolution of the earlier Moderators described in the previous section. The CM has been designed to extend the earlier moderator concepts for supporting knowledge based collaborations. A CM is capable of behaving differently for different types of users and therefore caters for the varying requirements of individual users depending on the roles they have in the collaborations. Earlier work on moderators concentrated on identifying actions of team members that are likely to cause problems for others. A CM makes a generalisation of this concept and defines the items that are of interest (IOIs) to individual partners and monitors for the events that correspond to any changes in these items. This generalisation captures the earlier objectives of moderators of identifying possible conflicts and at the same time extends the capabilities of the CM to monitor for external influences. The following example explains the advantages of this concept:

> ***Example of a typical moderation activity carried out by a CM:*** A component is manufactured by a certain partner and is then transported by a logistics partner. For the logistics partner, the IOIs include the dimensions of the component. Later the design partner decides on a better design of the component and agrees it with the manufacturing partner. The new design includes slight changes in the dimensions. As the dimensions are defined as one of the IOIs for the logistics partner, when a change in the state of a dimension is detected with the “design change” event; the logistics partner will be notified of the possible conflict between the component’s dimension and the logistics partner’s pallet size. In this way, the logistics partner is automatically made ***aware*** of the possible problem as soon as the design change is decided by the design partner

- It should be noted that in the above example, even if the dimensions were changed by the manufacturing partner, the same process would have resulted in raising the awareness of the logistics partner to the potential pallet size problems (due to the changed component dimensions). Therefore this approach also allows monitoring IOIs where changes are caused by actors that are unknown (at the time of definition of the IOIs).
- The CM allows the IOI to be an abstract item, e.g. opportunities to do a new business or form a new collaboration. The events affecting these IOIs can be internal to the collaboration or external e.g. publishing of a tender or entry of a new member in a collaboration pool. By monitoring events that affect the IOIs of users, the CM is able to raise awareness appropriately. It is worth mentioning that this approach makes it possible for the moderator knowledge to evolve. At the time of defining the IOI, a certain set of events may be responsible for affecting a certain IOI. However at a later stage, it may be realised that a new event also affects the same IOI. The moderation knowledge evolves by defining and adding the new event type to the knowledge base.

A high level description of the collaboration moderator service (CMS) and details of its architecture are provided by Swarnkar *et al.* (2011). In the following sub–sections, this paper describes the architecture and components of a CM from an implementation perspective. Two use cases for which the prototype CM was implemented as a service (Pre-Creation use case and Operational use case) are also mentioned in these sections.

3.1 CM Architecture

Figure 1 shows the core CM components as solid shapes. The user facing components (GUIs) can also be part of the CM; however this is not always the case as a CM can be deployed as a service to be used by other applications. Keeping this in mind, the CM has been designed with a modular architecture. The CM relies on two types of knowledge and information repositories: a knowledge base (CMKB) for storing moderation knowledge and a temporary storage of knowledge and information about processes currently in progress. Two important components of the CM are the knowledge acquisition module (KAM) and the Real Time Module (RTM), which are responsible for interacting with the user for maintaining the moderation knowledge and for real time monitoring activities respectively. The RTM has interfaces for accessing shared collaboration information and to publish/subscribe for events which drive the moderation activities.

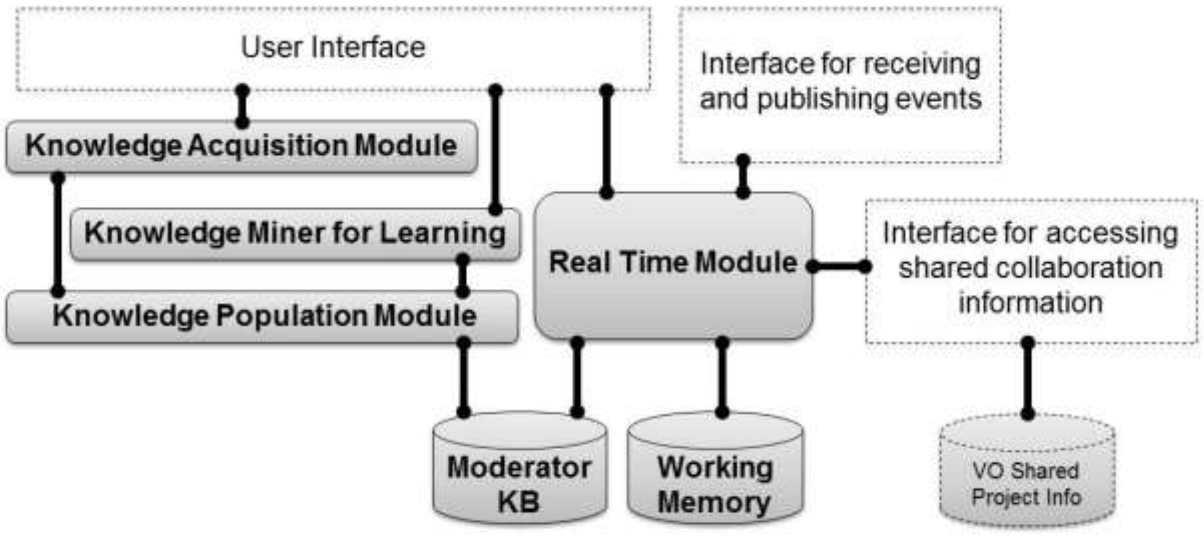

User Interface
Interface for receiving and publishing events
Knowledge Acquisition Module
Knowledge Miner for Learning
Real Time Module
Interface for accessing shared collaboration information
Knowledge Population Module
Moderator KB
Working Memory
VO Shared Project Info

FIGURE 1: COMPONENTS OF A COLLABORATION MODERATOR

3.2 CM Knowledge Acquisition Module

Knowledge acquisition is one of the major tasks for any knowledge based system and the CM is no exception to this. In order to perform the moderation tasks, the CM needs knowledge about its users. A user can have various roles in a collaboration and/or can be a part of various collaborations. Therefore the KAM needs to adapt itself to varying knowledge requirements for different 'avatars' of the same user. The moderation knowledge may include information about user's competencies, which can be either entered by the users themselves or can be extracted by the KAM using sources defined by the user. Similarly, if the moderation activity is about a process which is currently going on; the KAM may need to fetch information from the event logs or CM's working memory (WM – to be discussed later). Figure 2 shows an abstraction of the ontology underlying the CMKB. The ontology defines users to have multiple roles, and for each of these roles, there is an expert module (EM). An EM represents the moderation knowledge for a certain role of a certain user. The EM consists of a collection of units of moderation knowledge represented by IOIs. An instance of an IOI will contain the events it is triggered by, the conditions that analyse the event details (along with other sources of knowledge – like WM) and the actions that can be taken.

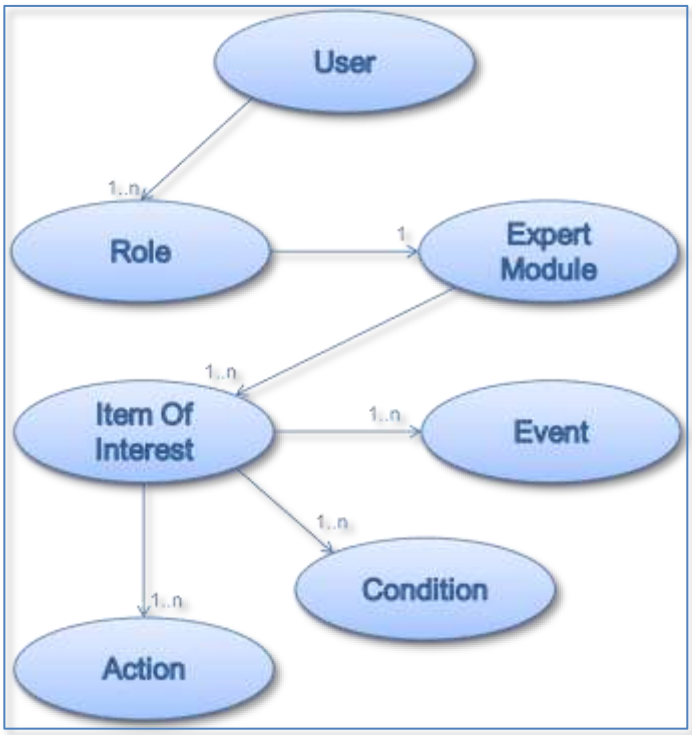


FIGURE 2: COLLABORATION MODERATOR KNOWLEDGE BASE ONTOLOGY

The process of knowledge acquisition is driven by relations present in a moderator ontology, which defines what sorts of roles are possible for a user. Each user has at least one role, i.e. as an individual. When a user joins a collaboration, the number of roles he is playing increases. In a service oriented environment, the CM detects the presence of roles associated with the user by querying the relevant service. The CM defines an EM for every role a user has, and each EM in turn will hold a number of IOIs.

It is important to note that the IOIs for a certain type of role may or may not be the same for another type of role. For example, a user as an individual may be interested in new collaboration opportunities; however the same user as a member of a VO will be interested in activities happening inside the VO. Therefore the types of IOI applicable for a role are also defined in the knowledge representation. For implementation, this can be encapsulated within the logic of the CM, however doing so will make the code highly coupled and very difficult to maintain. This poses a design challenge for the implementation of the knowledge base. In order to overcome this, the CM embraces the design principle: *program to an interface (super type), not an implementation*. Moreover, the *factory design pattern* (Gamma *et al.* 1995) has been used for creating the instances of knowledge units. Figure 3 shows the ontology defining a set of possible types of IOI that a role can be assigned. The various implementations of the superclass Role have a one–to–many "canHave" relation with appropriate implementations of the IOI class. Using the above design principle, the code is written for the superclass (in this case Role and IOI), therefore they can still work with the subclasses. When it comes to instantiation, factory methods are utilised to appropriately create objects. At some point during the course of VO operation, if a new type of IOI needs to be supported, it can be defined and associated with the type of role. The factory method will still be able to create objects as needed. In this way, the factory design pattern also ensures that the knowledge is ready for evolution.

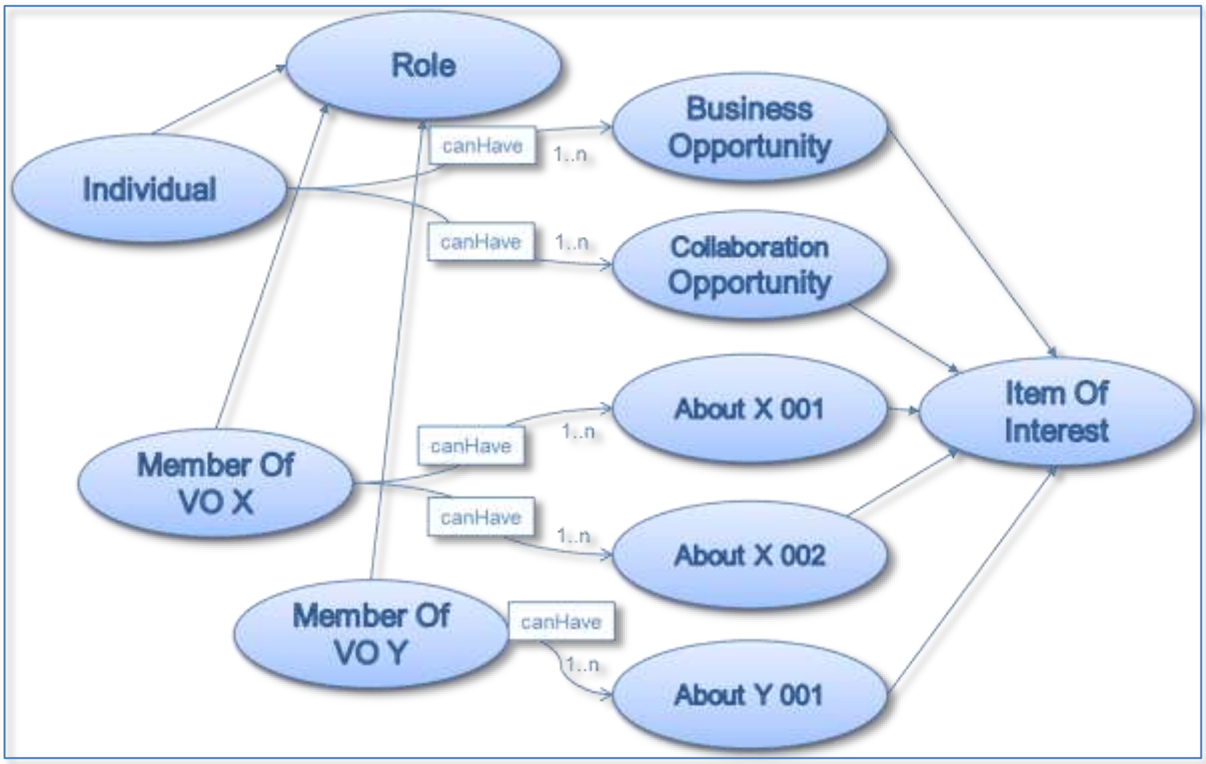


FIGURE 3: ONTOLOGY DEFINING TYPES OF IOI APPLICABLE FOR DIFFERENT ROLES

Further, an instance of an IOI must define what knowledge it requires from the user in order to carry out the moderation activities, i.e. how to populate and maintain information about events, conditions and actions. The IOI defines an interface to acquire knowledge, populate the KB, or execute actions etc., however it is the various subclasses which take care of these activities differently. In this way, the behaviour of the KAM is defined by the different IOIs being created or updated. The development of a user interface for the KAM demands great flexibility, therefore separating the presentation from the logic becomes very important.

3.3 CM Real Time Module

The Real Time Module (RTM) is responsible for carrying out the CM's on-going moderation activities. The RTM has access to the moderation knowledge and other sources of information to establish the context of the events occurring in real time and assess their impact on the users (i.e. through the IOIs). The RTM is initially in a monitoring state, listening to subscribed events to detect those which may affect one or more of CM's users. When such an event is detected, the RTM enters the assessment mode and analyses the situation. This may involve retrieving additional information and knowledge from external and internal repositories. Depending on the assessment, the RTM takes appropriate action and then returns to its initial idle/monitoring state.

Figure 4shows an instance of a monitoring process carried out by the CM. When the event arrives from an event publisher, there is a message payload attached to it. The message payload can contain textual information and files as an attachment while the event itself contains a variety of information like topic, publisher details etc. The RTM parses this information and matches the type of event with possible IOIs that may be affected by this event. When one or more matching IOIs are found, the RTM executes them and they carry out further analysis of the conditions and take appropriate actions.

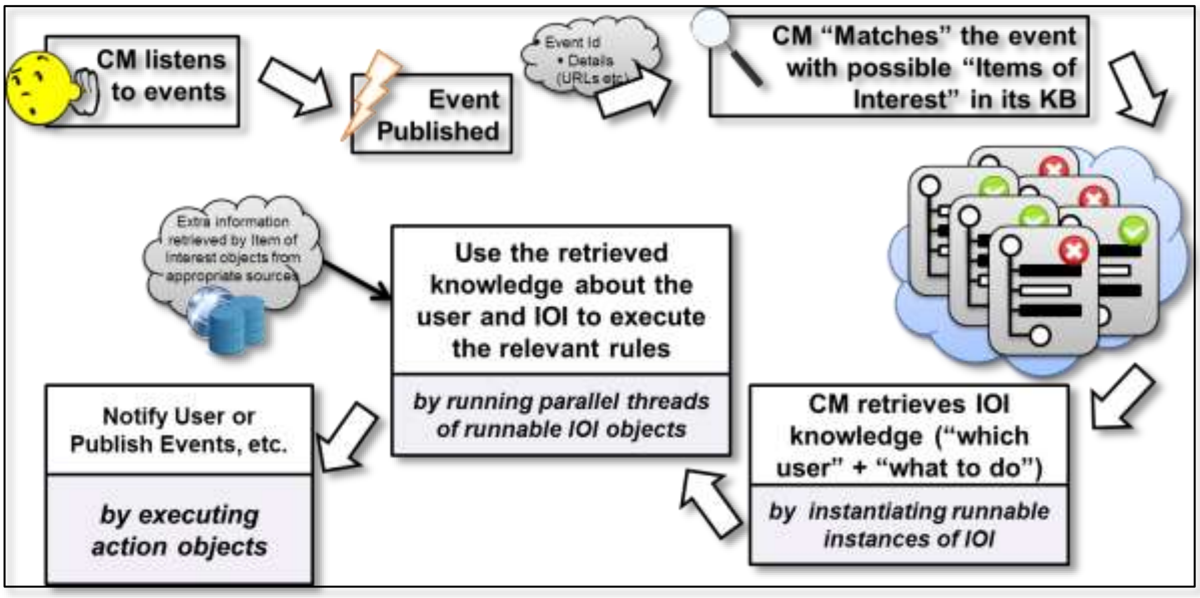

CM listens to events
Event Published
• Event Id
• Details (URLs etc)
CM "Matches" the event with possible "Items of Interest" in its KB
Extra information retrieved by Item of Interest objects from appropriate sources
Use the retrieved knowledge about the user and IOI to execute the relevant rules
by running parallel threads of runnable IOI objects
Notify User or Publish Events, etc.
by executing action objects
CM retrieves IOI knowledge ("which user" + "what to do")
by instantiating runnable instances of IOI

FIGURE 4: MODERATION PROCESS CARRIED OUT BY CM

It is understandable that an instance of the moderation process, as shown in figure 4, may not finish before a new event occurs. This can happen in very ordinary situations, where the process life cycle lasts for a few seconds. If the RTM had to wait until a current moderation process finishes before it can start another, the CM would be rather useless for all practical purposes. Therefore any implementation of the RTM needs to be multi–threaded. Figure 5 shows the steps of the moderation process in detail, taking care of concurrent processing aspects. As the RTM's monitoring thread (steps shown in purple with solid lines in the figure) receives an event, it generates a query thread (steps shown in orange with dashed lines in the figure). The original monitoring thread goes back to its initial listening state immediately. It is then the newly created thread that queries the KB for IOI that are affected by the class of the received event. If it retrieves IOIs that are likely to be affected, it creates parallel threads (steps shown in green with dotted lines in the figure) for assessment of the event with respect to moderation knowledge contained in each of the IOIs. The parallel assessment threads carry out the analysis. They may need to retrieve additional information from external or internal sources. After the analysis, appropriate actions are taken. If the assessment results in multiple actions, they may be taken by the creation of yet another set of threads (shown in blue with solid lines at the bottom of the figure).

However, it should be noted that concurrency comes with the cost of complexity. The KB is most likely to be queried by the multiple threads at the same time; therefore the interface to the KB should be able to handle concurrent queries. At later stages, when the IOIs carry out the assessment, they may need to access the WM concurrently, which may include all of the four basic functions of persistent storage: create, read, update and delete (CRUD) operations. Therefore, the implementation of the RTM should take care of the fact the concurrent threads will be sharing resources and therefore the development needs to be thread–safe.

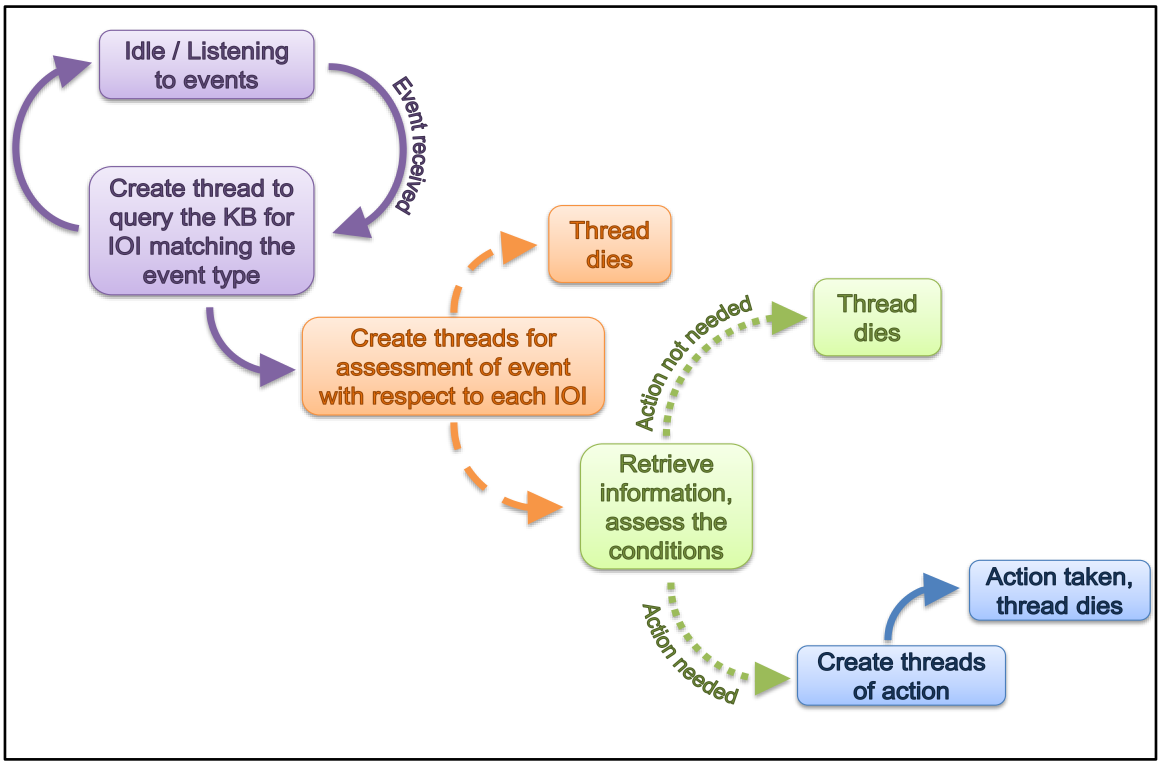

Idle / Listening to events
Event received
Create thread to query the KB for IOI matching the event type
Thread dies
Create threads for assessment of event with respect to each IOI
Action not needed
Thread dies
Retrieve information, assess the conditions
Action needed
Create threads of action
Action taken, thread dies

FIGURE 5: DETAILED STEPS IN THE MODERATION PROCESS

3.4 Moderator Knowledge Base

The moderator KB contains knowledge about CM users. The abstract ontology of a CM user profile (Figure 2) was discussed in an earlier section. The KB stores an instance of user profile for each CM user, and each user profile contains an EM corresponding to each role that the user has and zero–to–many IOIs are associated with each EM. The KB provides an interface for the KAM to populate and modify the knowledge, while the RTM is querying it. The way these modules operate and access the knowledge creates two very different views of the same knowledge as shown in the following figures (Figure 6 and Figure 7). Figure 6 shows a conceptual view of how the KB appears to the KAM. Each user profile is a tree with the User node at the root and with various nodes as EMs or IOIs. However, when the RTM accesses the KB, its focus is on querying the events that it has received. Figure 7 shows another conceptual view of the KB as it appears to the RTM, where the Events are the root nodes, with others like IOI and EM attached to it. The KB is implemented as a graph based knowledge repository with semantic annotations.

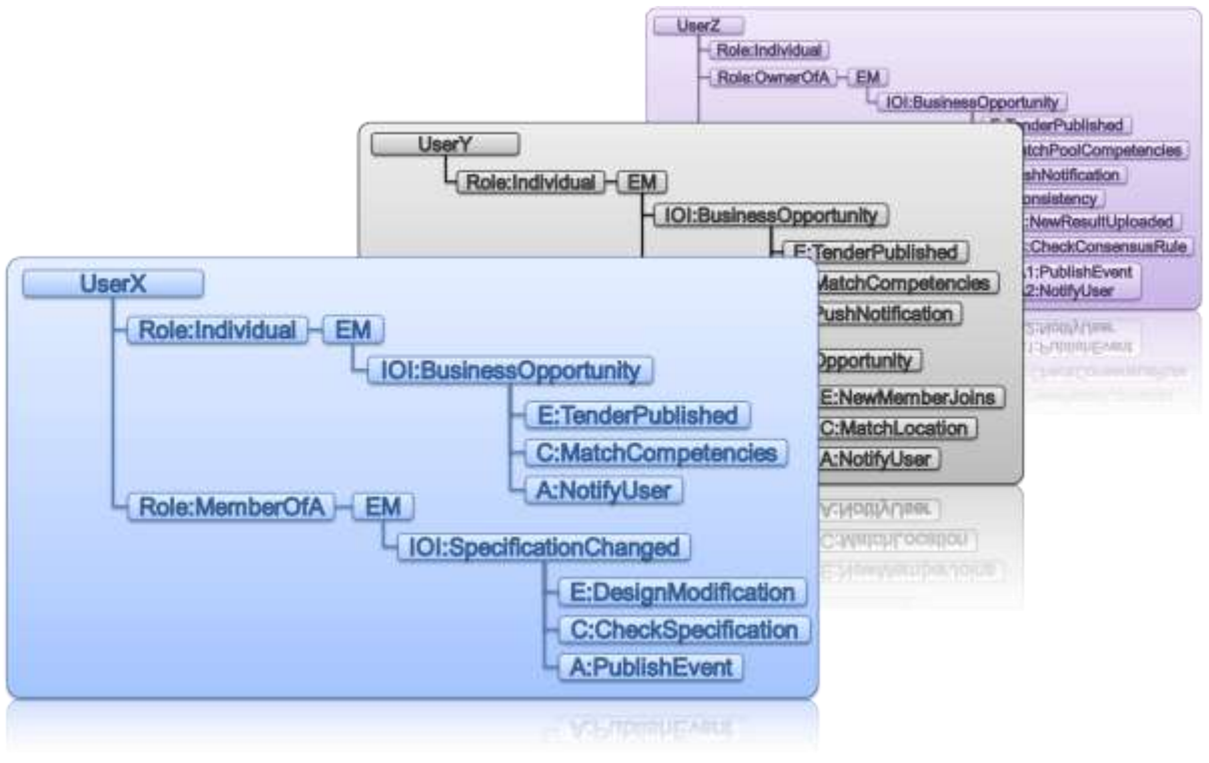


FIGURE 6: KAM VIEW OF THE KNOWLEDGEBASE

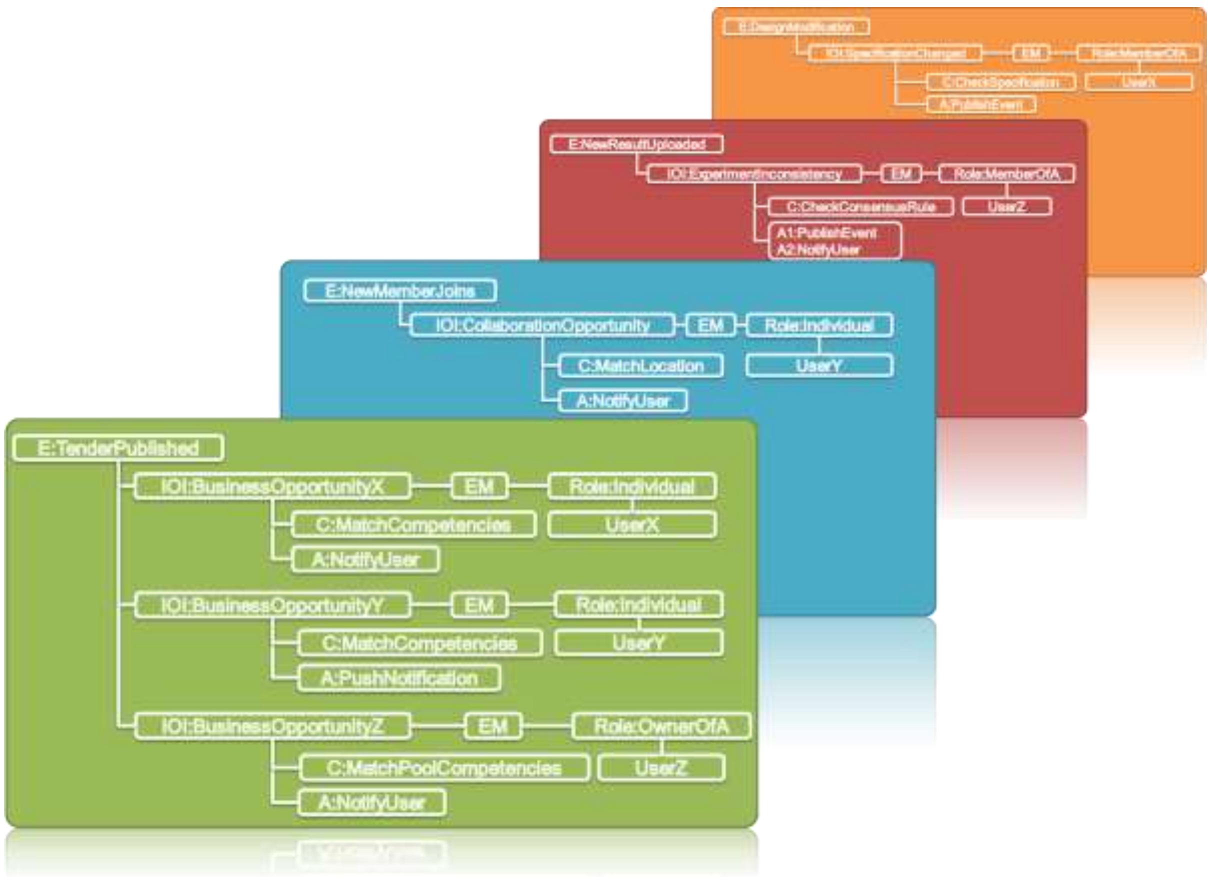


FIGURE 7 : RTM VIEW OF THE KNOWLEDGEBASE

3.5 WORKING MEMORY

The KB is a repository of moderation knowledge and for practical purposes, the RTM has read–only access to it. There is a possibility for the RTM to feedback "lesson–learnt" type knowledge to the KB in run time, however that should happen through a Knowledge Population Module, thereby leaving the RTM with a read–only direct access (Swarnkar *et al.* 2011). A moderation activity may involve analysing multiple events in succession over a certain period of time. Therefore the RTM needs a means of persisting such intermediate knowledge of "what's going on" – a way of reading/writing the information relevant to the current moderation processes. The CM achieves this through Working Memory (WM) which can be defined as *"a programming interface that allows transient storage, retrieval and traversal of events via a managed list in real time"*. Figure 8 shows how the WM is utilised by the RTM.

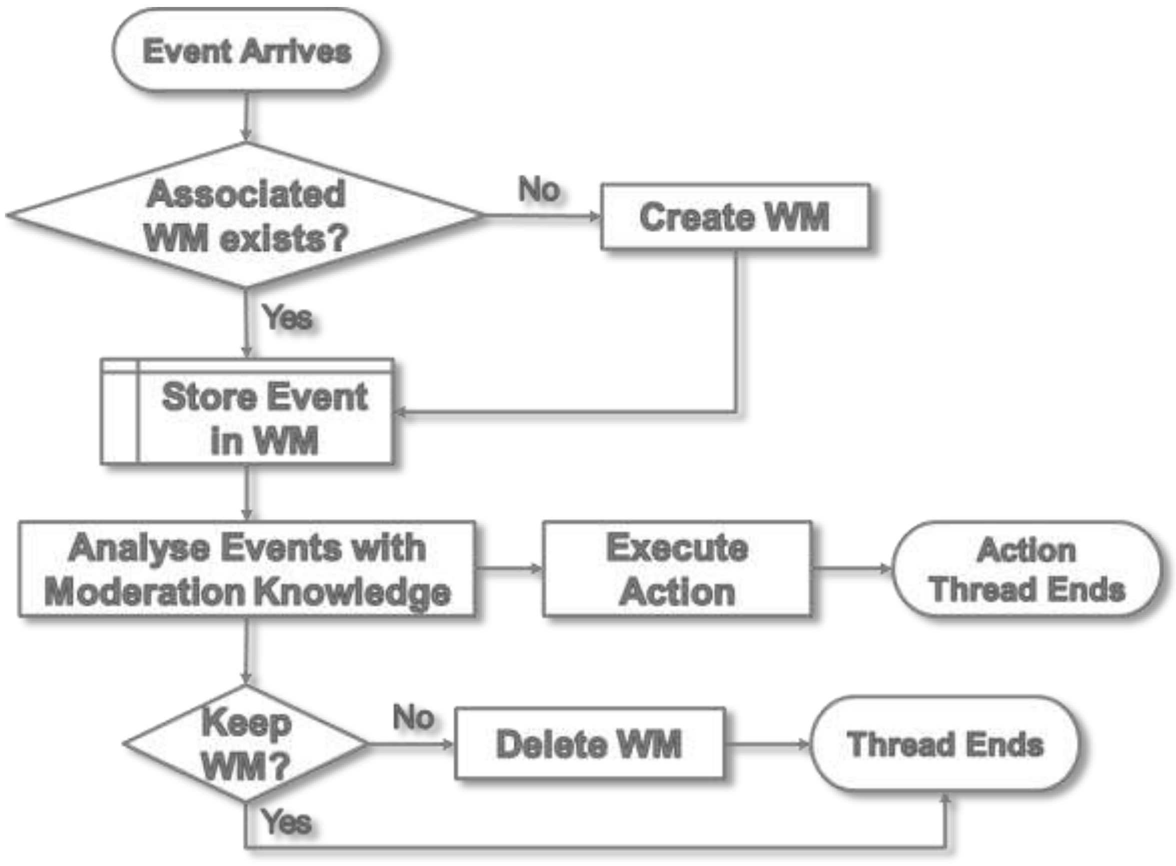


FIGURE 8: TYPICAL FLOW DIAGRAM SHOWING THE USE OF WORKING MEMORY IN THE RTM

The WM class provides methods for getting an instance of the WM, manipulating the event list and finally deleting the persistence, once the activity has finished. The *factory design pattern* was used again as it separates the logic of creating the instances of WM from the moderation code and makes maintenance issues easier to handle.

It is understandable that different implementations of the WM will have different requirements. The implementation can be memory based, file based or database based. It may appear lucrative to use a memory based persistence as it can be more efficient to implement for complicated objects, however if it is expected to contain a large number of events, it may be better to use a low latency lightweight relational database. On the other hand, an object–oriented database may prove to be the option which provides the best of both worlds. A smart way of developing, keeping these considerations in mind, is to utilise the *strategy pattern* (Gamma *et al.* 1995) and use different types of persistence implementation for different situations.

Figure 8 also shows the flow of logic for getting the WM instance. There is a specialty of the WM instances – i.e. for one moderation activity, there should be one and only one instance of the WM. When the event arrives, the RTM looks for an existing WM associated with the moderation activity (i.e. if the moderation activity has already begun). If found, the existing instance is used otherwise a new instance is created. Therefore, in most cases the WM implementation should be done with the *singleton pattern* (Gamma *et al.* 1995).

4 Collaboration Moderator Service

The work reported in this paper has been carried out as part of the SYNERGY project (SYNERGY 2008). SYNERGY was an FP7 collaborative research project between 8 academic and industrial partners across 5 European countries and was funded by the European Commission. The project ran for over 3 years with the aim of developing dynamic and adaptive knowledge management systems and services to enable VOs to collaborate more easily. The partners in the SYNERGY project worked towards developing a web–based and service oriented software infrastructure to help all kinds of companies which need to engage in collaborative business, to discover, capture, deliver and apply knowledge relevant to collaboration creation and operation thus helping them to effectively and efficiently participate in VOs whilst avoiding the above mentioned shortcomings and problems. The projects utilised an enterprise service bus (ESB), service oriented architecture (SOA) and event driven architecture (EDA) as well as web-service notification (WSN) as enabling technologies (SYNERGY Consortium 2008).

The Collaboration Moderator Service (CMS) was developed as one of the services offered by the SYNERGY platform. Work was carried out to identify, with end-user partners and others, the range, scope and content of knowledge relevant to collaborative working, including the identification and abstraction of collaboration patterns for the learning enterprise and VO. Keeping in mind the target users, i.e. small to medium sized enterprises (SMEs), the requirement of a software–as–a–service model was realised. Moreover, CMS was developed as an interoperable service, which can be utilised by other services or applications in a SOA environment as shown in Figure 9.

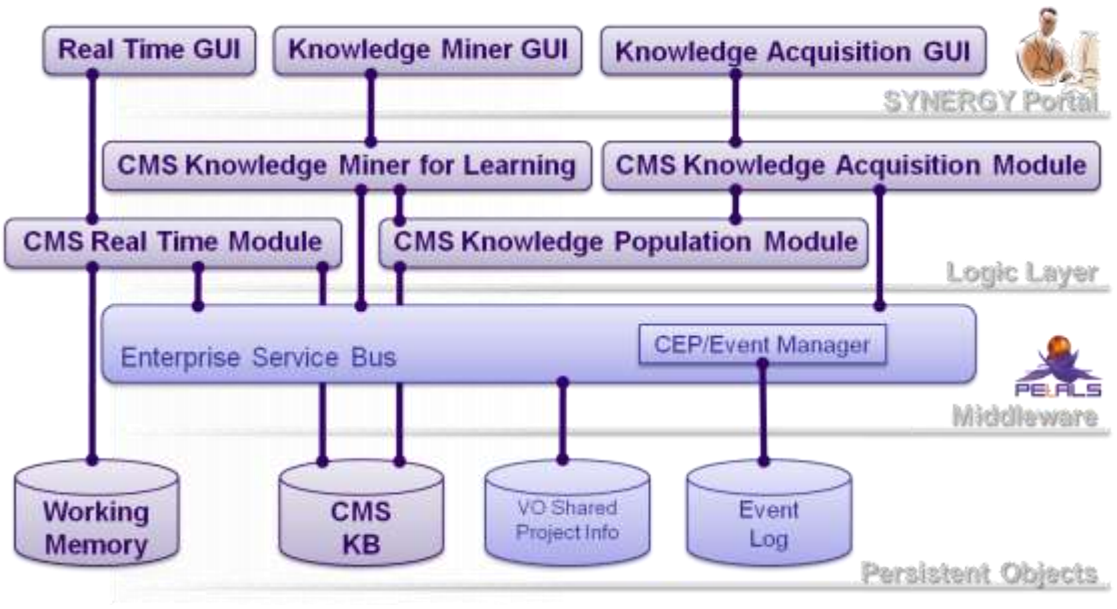


FIGURE 9: CMS ARCHITECTURE IN CONTEXT OF SYNERGY

5 Use Cases

During the development of the prototype CMS, two types of use cases were considered. The use cases were selected to aid the VO initiation and operational phases. The use cases exhibit how CMS can be useful to two very different kinds of end users involved in completely different types of activities. Also, for both of the use cases, CMS is provided as a service on the ESB. In one case, it was provided to the end user on a software–as–a–service (SaaS) model, while in the other case it is provided as a business–to–business (B2B) service to be used by the exploiter. The following sections present the storylines behind the development of each use case and explain how CMS aids them.

5.1 SMECluster Use Case

The main actor in this use case is a facilitator for SMEs working in the automotive area. It has a database with the information of 300+ competent enterprises relevant to the field. The objective of the facilitator is to search for new business opportunities and assess their suitability with regards to the existing pool of enterprises. If competent enterprises are found, the facilitator then initiates an assessment of possible VOs considering risk factors etc. and subsequently proposes a business proposition to the members. The facilitator uses the web based SMECluster portal for its business.

The objective of the SMECluster interface is to serve both the VO facilitator and the member enterprises, however the instances are quite different in both cases. The VO facilitator is presented with tools to view new business opportunities, create a new business opportunity, provide suggestions on the potential VO members and the risks associated with them. On the other hand, the member enterprises are shown the opportunities suggested through the facilitator. For both the facilitator and the members, the interface provides ways to accept or reject suggestions.

5.1.1 CMS in the context of SMECluster use case

For the pilot study on this use case, the available database of 300+ relevant automotive enterprises was text mined and the competency knowledge was extracted. This was stored using the ECOS structure (Khilwani *et al.* 2011) and made available for queries by a knowledge management service within SYNERGY. Figure 10 shows the process of converting the company data in legacy format to semantically annotated competency knowledge. A collaboration pool is created and the companies which are served by SMECluster are added. A collaboration pool (CPOOL) is a collection of enterprises which are willing to collaborate for suitable business opportunities.

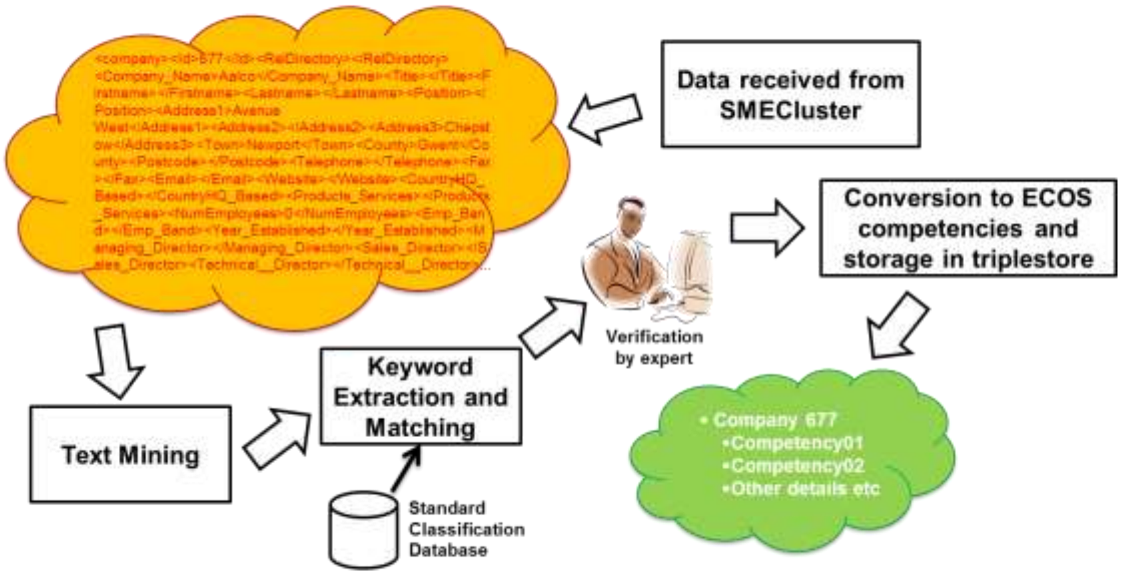

<company><id>677</id><RelDirectory></RelDirectory><Company_Name>Aalco</Company_Name><Title></Title><Firstname></Firstname><Lastname></Lastname><Position></Position><Address1>Avenue West</Address1><Address2></Address2><Address3>Chepstow</Address3><Town>Newport</Town><County>Gwent</County><Postcode></Postcode><Telephone></Telephone><Fax></Fax><Email></Email><Website></Website><CountryHQ_Based></CountryHQ_Based><Products_Services></Products_Services><NumEmployees>0</NumEmployees><Emp_Band></Emp_Band><Year_Established></Year_Established><Managing_Director></Managing_Director><Sales_Director></Sales_Director><Technical__Director></Technical__Director>...
Data received from SMECluster
Conversion to ECOS competencies and storage in triplestore
Verification by expert
Text Mining
Keyword Extraction and Matching
Standard Classification Database
• Company 677
•Competency01
•Competency02
•Other details etc

FIGURE 10: PREPARING THE SHARED KNOWLEDGEBASE FOR VO INITIATION USE CASE

CMS needs to acquire moderation knowledge before it can help its users. When the facilitator registers with CMS, the KAM queries the knowledge management service to find out what types of roles the facilitator has and creates EMs accordingly in the KB. Now the facilitator creates an IOI of the type "business opportunity". At this point, CMS subscribes to events associated with the occurrences of new business opportunities. These can come from either custom built search engines as they fetch public tenders or by manual entry of opportunities. These are provided as interoperable services on the ESB, and therefore it does not matter to CMS where they come from as they conform to the pre–agreed event schema. The EM corresponds to the facilitator role; therefore the KAM created the IOI using knowledge analysis that needs to be done with respect to the entire CPOOL.

Upon receiving these events, the CMS analyses the opportunity and extracts the competency requirements. The RTM is triggered and it looks for IOIs associated with the event type. The IOI contains the moderation knowledge on what sorts of capabilities to look for and what range of enterprises to explore, which in this case is the set of all the competencies for each member in the entire pool. The available pool of competencies is extracted by querying the shared knowledge base. A further analysis is done which identifies which combinations of the companies can be expected to fulfil the requirements the best. It creates a superset of all the possible combinations of enterprises for the requirements and then analyses their suitability, thereby ranking them. For example, the combinations of potential candidates for VO formation with fewer candidates are ranked higher.

The action in this case is sending the result of analysis to be received by SMECluster, which is done via raising an event containing the results as a message payload back to the ESB. CMS generates the event with a message payload with two parts: first, a textual part which is written in plain English, describing the analysis results briefly, and second, the itemized information in a pre–agreed schema. SMECluster upon receiving this event from the petals bus utilises the information provided to present the facilitator with the suggestions and options for forming a new VO and applying various tools like risk assessment etc. (Alawamleh & Popplewell, 2011)

5.1.2 Manual Entry of Business Opportunity

In this case, the facilitator receives a business opportunity from its own professional network. In most cases, these are in the form of a telephone conversation accompanied by a free text email explaining the requirements. As mentioned before, the SMECluster portal provides the facilitator with options for creating new business opportunities. The facilitator uses the SMECluster interface to create the business opportunity on the system. SMECluster then generates an event with the requirements as a message payload and publishes it to the ESB, which in turn will send it to CMS (subscriber). This triggers the analysis process at CMS as described above.

Automated Search for Business Opportunities

This functionality of CMS is there for automatically fetching suitable business opportunities for the facilitator with added support for creation of new collaborations; however CMS is not a search engine. This functionality is achieved with the same publish subscribe mechanism mentioned above. CMS acts on receiving business opportunity events – such events can be produced by a search program that can be built with different parameters according to the implementation requirements. In this case, a web bot was written which fetches new business opportunities from predefined sources. When it finds a new opportunity, it extracts the information and publishes an event with the information as the message payload. For CMS, it is the same event containing a business opportunity and it triggers the CMS in the same way as discussed in the previous section. The analysed business opportunity will appear on the facilitator's screen used by SMECluster.

5.2 SAM Use Case

The actors involved in this use case are the project manager and other members of an operational VO. The lead organisation (DC) setup a drug discovery called Scientists against Malaria (SAM) with the aim of conducting computational prediction experiments for protein targets. The nature of the work is collaborative, i.e. different experiments conducted by different partners, with experiments having dependencies on those carried out at other sites. Moreover the trends in the results also affect the overall strategy and success of the project.

The various steps of drug discovery involve comparisons and discussions of the predictions of different modelling and design methods and additionally an evaluation of experimental results. The VO uses the Collaborative Electronic Research Framework (CERF) software which provides a web-based collaborative lab notebook to the project. CERF is integrated with SYNERGY services for exchanging information in a schema that was developed by DC.

The various members in the VO perform different roles, and different prediction experiments, screening and assays happen at different locations. As the number of experiments is in excess of thousands, it becomes very difficult for the project manager and the VO members to determine if there are some inconsistent results or erroneous estimates.

The main interface for these VO members is the electronic lab notebook CERF, where all the data entry (prediction or assay results) takes place. CERF provides the VO members and project manager with an interface to enter and view the experimental results in the form of a dashboard. Based on the results of predictions, a consensus is reached whether to proceed with more time consuming, sophisticated and expensive assays for compounds. The consensus rule decision on the predictions for a compound is shown as traffic lights: green means the compound passes, red means that it fails while yellow represents an inconclusive result. This provides the members involved in experiments with a subset of the entire collection of compounds to carry out the expensive assays on.

At the next stage, the member carrying out the assays enters the results for the compounds as they finish each individual experiment. The assay results which match with the predictions show a positive outcome. However there are situations where the results are inconsistent with the predictions, and the project manager would like to know if there are many such inconsistencies.

5.2.1 CMS in the context of SAM pilot

When individual members enter results in CERF, it sends the data as a schema-based instance message to the Petals service bus, which in turn distributes the data as event messages to the SYNERGY services, including CMS. CMS also receives events triggered by a Complex Event Processing Engine (CEPS) based on different patterns in the data. Unlike, the other use case, here the users are provided with a user interface where they can interact with CMS. An important aspect of this pilot as far as CMS is concerned is that it provides an example of how the KAM can be customised to meet the needs of a particular user type and thereby add value to their collaboration activities.

Generation of Consensus Rules – specialised KAM

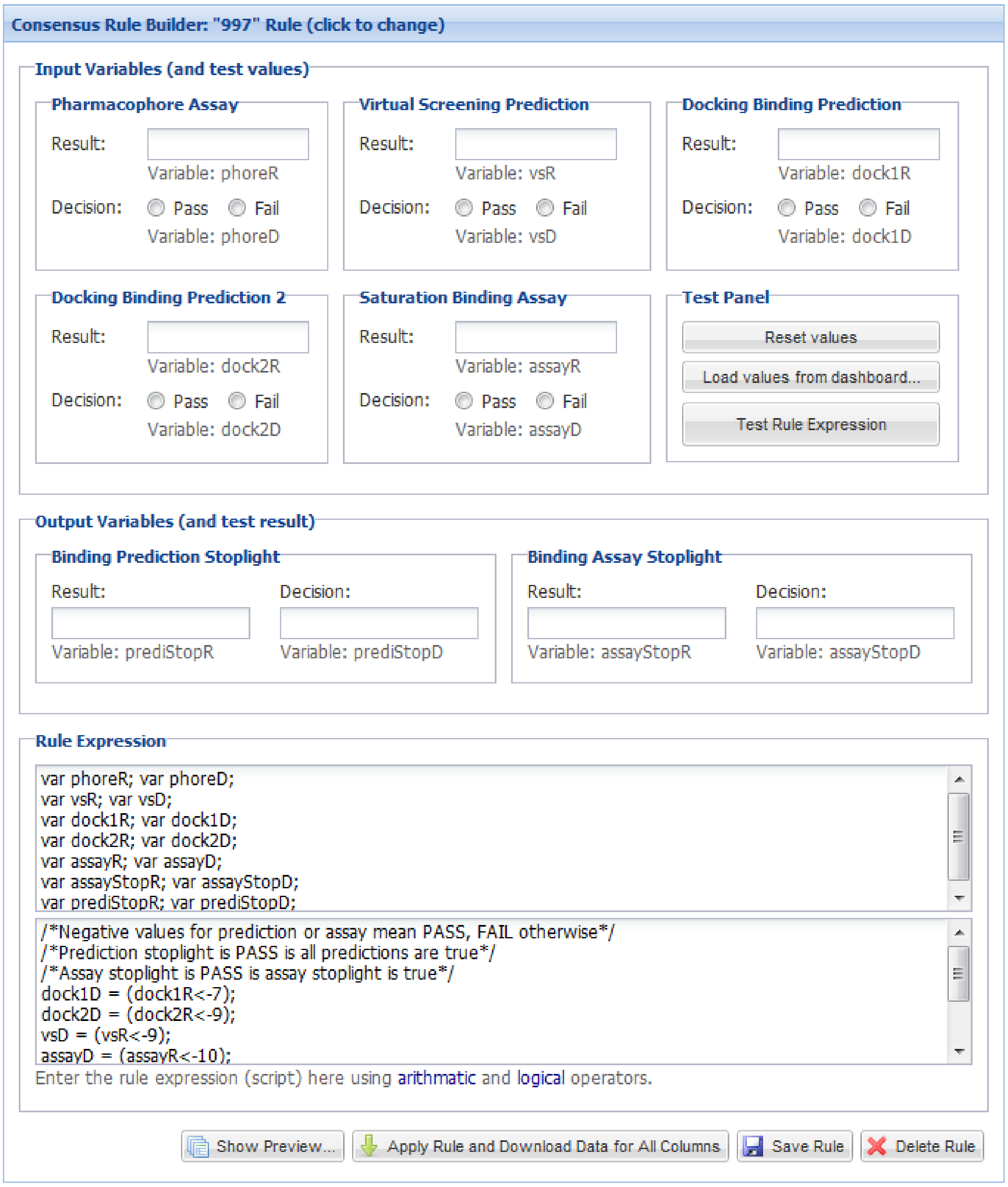
For creating IOIs related to the experimental dashboard, the users need to create rules that they can understand and work with the results as they arrive in real time. At the same time, the user should be able to create and see the effects of the rules in an interactive and easy way before they apply them. The rules thus created will be executed by CMS at runtime on the event data. The rules need to be written in a generic way, so that they can be applied to any column on the dashboard and when attached to the IOI, be executed in real time with respect to event messages associated with those columns.

In predictive drug discovery, it is a general practice to screen a smaller set of compounds for further experiments as the team progresses to more expensive and time consuming experiments. The screening is based on the outcome of the experiments that have already been carried out on the compound. In the SAM use case, after the prediction and screening results arrive on the dashboard, the VO project manager needed to select a smaller set of compounds based on a consensus rule. This much smaller selection consists of the candidate compounds for further assay experiments. It was realised that the rules generated for moderation activities can very well serve as the consensus rules on the existing data. Therefore work was carried out to specialise the KAM to accommodate this requirement of the pilot.

In the operational stage of the VO, CMS subscribes to the relevant event topics so that it can receive events associated with prediction or assay result registration at CERF. In this way, CMS is able to create a copy of the result data in its temporary storage. Apart from the actual result, the experiment/assay/prediction result registration events contain information about the compound, the investigator, the timestamp etc. The project manager is provided with the specialised KAM and presents its users with an interface to work with CMS's copy of the dashboard, and to generate knowledge related to possible content which may appear in the CERF dashboard. Hence, using the specialised KAM, the project manager and other VO partners can create consensus rules to be applied to the existing dashboard data. The specialised KAM allows its user full flexibility to manipulate the data, ability to check its validity on a sample from the dashboard or ability to apply rules on the whole data set using a rule builder interface (Figure 11). The consensus rules themselves are in the form of scripts and therefore they can be stored normally as data. Hence, the user can also store the rules generated to use later. The VO members may need to apply different consensus rules on the existing dashboard and see how they affect the resulting columns and when they are happy with the results; they can extract the data with consensus rules applied to it. This data goes back to CERF as prediction and assay spotlights. The rule builder interface allows full flexibility in the creation of the consensus rules, i.e., whether to base the rule on the prediction rule outcomes, the underlying real values, or both. To build the consensus rule, a number of variables are pre-defined as shown in Table 1.

TABLE 1: VARIABLES USED IN THE RULE BUILDER

Variable name	Meaning
phoreR, phoreD	Value (phoreR) and PASS/FAIL status (phoreD) of the "Phore" column
dock1R, dock1D	Value and PASS/FAIL status of the "DOCK" column
dock2R, dock2D	Value and PASS/FAIL status of the "DOCK 2" column
vsR, vsD	Value and PASS/FAIL status of the Virtual Screening column
prediStopR; prediStopD	Value and GREEN/RED status of Binding Prediction Stoplight
assayR, assayD	Value and PASS/FAIL status of the Binding Assay column
assayStopR, assayStopD	Value and GREEN/RED status of Binding Assay Stoplight


Consensus Rule Builder: "997" Rule (click to change)

Input Variables (and test values)

Pharmacophore Assay
Result: Variable: phoreR
Decision: ◯ Pass ◯ Fail Variable: phoreD

Virtual Screening Prediction
Result: Variable: vsR
Decision: ◯ Pass ◯ Fail Variable: vsD

Docking Binding Prediction
Result: Variable: dock1R
Decision: ◯ Pass ◯ Fail Variable: dock1D

Docking Binding Prediction 2
Result: Variable: dock2R
Decision: ◯ Pass ◯ Fail Variable: dock2D

Saturation Binding Assay
Result: Variable: assayR
Decision: ◯ Pass ◯ Fail Variable: assayD

Test Panel

Output Variables (and test result)

Binding Prediction Stoplight
Result: Variable: prediStopR
Decision: Variable: prediStopD

Binding Assay Stoplight
Result: Variable: assayStopR
Decision: Variable: assayStopD

Rule Expression

```
var phoreR; var phoreD;
var vsR; var vsD;
var dock1R; var dock1D;
var dock2R; var dock2D;
var assayR; var assayD;
var assayStopR; var assayStopD;
var prediStopR; var prediStopD;
```

```
/*Negative values for prediction or assay mean PASS, FAIL otherwise*/
/*Prediction stoplight is PASS is all predictions are true*/
/*Assay stoplight is PASS is assay stoplight is true*/
dock1D = (dock1R<-7);
dock2D = (dock2R<-9);
vsD = (vsR<-9);
assayD = (assayR<-10);
```

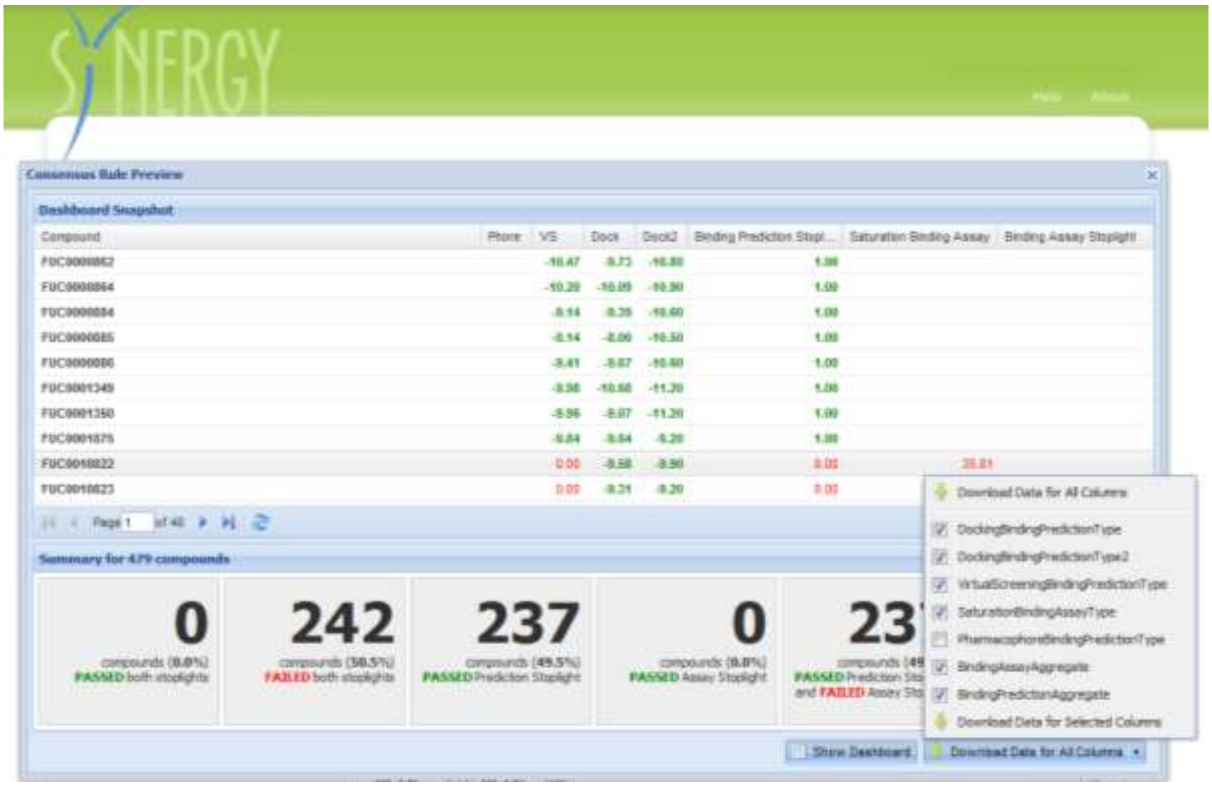
Enter the rule expression (script) here using arithmatic and logical operators.

Reset values
Load values from dashboard...
Test Rule Expression
Show Preview...
Apply Rule and Download Data for All Columns
Save Rule
Delete Rule

FIGURE 11: THE CONSENSUS RULE BUILDER AT CMS KNOWLEDGE GENERATOR

Consensus rules are designed in a text-based manner, using these variables as well as additional, user-defined ones. The rules defined can apply to any column. In order for the users to test and execute the rules, the specialised KAM needs a copy of the dashboard data it receives from events. As it populates itself from (possibly) concurrent events, the dashboard persistence for the specialised KAM draws inspiration from the WM discussed earlier. This has been implemented as an object database. Each row object (representing all the results for a compound) contains compound identifiers and a collection of individual results objects. This object is able to execute the consensus rule script and generate aggregates (results for prediction spotlights).

Once a consensus rule is designed, the user is presented with a preview of the consensus rule on the dashboard to show the statistics on the outcome of the consensus rule. The user can also see a snapshot of the dashboard with the rule applied on it and download the data in a format that can be directly uploaded to CERF (Figure 12).



Consensus Rule Preview

Dashboard Snapshot

Compound	Phore	VS	Dock	Dock2	Binding Prediction Stopl...	Saturation Binding Assay	Binding Assay Stoplight
FUC0000862		-10.47	-8.73	-10.80	1.00		
FUC0000864		-10.20	-10.09	-10.90	1.00		
FUC0000084		-8.14	-8.20	-10.60	1.00		
FUC0000085		-8.14	-8.00	-10.50	1.00		
FUC0000086		-9.41	-9.67	-10.60	1.00		
FUC0001349		-8.98	-10.68	-11.20	1.00		
FUC0001350		-8.96	-8.07	-11.20	1.00		
FUC0001875		-8.84	-8.84	-8.20	1.00		
FUC0018822		0.00	-8.58	-8.90	0.00	35.81	
FUC0018823		0.00	-8.21	-8.20	0.00		

Page 1 of 48

Summary for 479 compounds

0	242	237	0	23
compounds (0.0%) PASSED both stoplights	compounds (50.5%) FAILED both stoplights	compounds (49.5%) PASSED Prediction Stoplight	compounds (0.0%) PASSED Assay Stoplight	compounds (49 PASSED Prediction Sto and FAILED Assay Sto

Download Data for All Columns
☑ DockingBindingPredictionType
☑ DockingBindingPredictionType2
☑ VirtualScreeningBindingPredictionType
☑ SaturationBindingAssayType
☐ PharmacophoreBindingPredictionType
☑ BindingAssayAggregate
☑ BindingPredictionAggregate
Download Data for Selected Columns
Show Dashboard
Download Data for All Columns

FIGURE 42: THE "PREVIEW AND APPLY RULE" WINDOW

Inconsistency of Experimental Results

The VO carries on performing the more expensive assay experiments with the set of compounds determined by the consensus rule. Ideally at this stage, it would be expected that all the results of the assays match with the aggregates (results of consensus rules for each compound). However, this does not always happen in real life. When the results do not match, the project manager or other concerned VO members would like to know about it, so that they can initiate activities to resolve the issue.

After applying the consensus rules on the set of prediction results, the project manager uploads the data to the CERF which sends the data as prediction aggregate registration events. At a later stage, when CERF sends the assay results, it is checked by matching them against the aggregates (value found by applying the consensus rule for the same compound). If the results do not match, this is something that the member involved in experiments needs to know. In this way, every time an inconsistency occurs, the concerned VO member is notified about it. In the use case, the occurrence of inconsistency was also notified by raising an "inconsistency data" event to the ESB.

CMS goes beyond the scenario presented above. The above serves well for a small number of experiments, however in reality with thousands of compounds; this may become a problem for the VO members. Keeping in mind that there are numerous compounds involved in the assay experiments, therefore the concerned VO members may not like to receive an email or a notification every time an inconsistency occurs. Depending on how important the assay is, the VO members may prefer to be notified when a certain number of inconsistencies have occurred within a certain period, or when a certain period has passed after the occurrence of the first such inconsistency. In some cases, they may wish some other sort of action to be taken in a different situation, e.g. to arrange a meeting when a certain number of inconsistencies happen in a given time; or simply to be notified by email when the number of inconsistencies is less than a certain number.

Along with the registration of aggregate results, the CMS also subscribes to the inconsistent data events. The CMS allows the user to define its IOI based on the inconsistency in the assay results. At the same time, the user can define what sort of actions they wish to be taken when different conditions are satisfied. The inconsistency events trigger the RTM, which looks for these IOIs in the KB. Once activated, these IOIs run as separate threads waiting for the conditions to be fulfilled. As more of such events occur, CMS analyses them along with the existing events in WM against the moderation knowledge and decides to either append the event to the working memory or delete the chain of events and take certain CMS actions, Also, if the same event is not received in a certain period after the occurrence of the first event, the event is considered to be timed out and a certain CMS action is taken. Once again, the moderation knowledge enables RTM to take appropriate actions in different situations. For example, if the number of inconsistencies is within a given limit, it could be that an email report is sent to the VO, on the other hand if the number of inconsistencies exceed the normal limit within certain time period, a CMS event may be produced and sent to the ESB, which in turn will initiate a collaboration pattern for starting a meeting.

This paper has reported on the design and prototype implementations of CMS based on two use case scenarios. The CMS is an evolution founded on several years of Moderator research.

ACKNOWLEDGEMENT

The authors thank the funders and researchers from the SYNERGY - Supporting highly adaptive Network Enterprise collaboration through semantically-enabled knowledge services. 7th Framework Prog - Grant Agreement No: 216089, for their support in this research.

REFERENCES

Alawamleh, M., Popplewell, K., 2011, Interprettive structural modelling of risk sources in a virtual organisation, International Journal of Production Research, 49:20, 6041-6063, doi: 10.1080/00207543.2010.519735

Camarinha-Matos, L.M., Oliveira, A.I., Sesana, M., Galeano, N., Demsar, D., Baldo, F., Jarimo, T., 2009, A framework for computer-assisted creation of dynamic virtual organisations, International Journal of Production Research, 47:17, 4661-4690, DOI: 10,1080/00207540902847272

Camarinha-Matos, L.M. (ed.) 2004. Virtual Enterprises and Collaborative Networks: IFIP 18th World Computer Congress : TC5/WG5.5 - 5th Working Conference on Virtual Enterprises, 22-27 August 2004, Toulouse, France

Camarinha-Matos, L.M.; Afsarmanesh, H., 1999, Infrastructures for Virtual Enterprises, KluwerAcademic Publishers, ISBN 0792386396, http://dare.uva.nl/record/80752

Camarinha-Matos, L. M., Afsarmanesh, H., Garita, C., Lima, C., 1998, Towards an architecture for virtual enterprises, Journal of Intelligent Manufacturing, Springer Netherlands, ISSN 0956-5515, 9 (2) pp 189 - 199, http://dx.doi.org/10.1023/A:1008880215595

Harding, J. A. and Popplewell, K., 1996. Driving concurrency in a distributed concurrent engineering project team: a specification for an Engineering Moderator, International Journal of Production Research, 34 (3), pp. 841-861, doi:10.1080/00207549608904937, http://www.tandfonline.com/doi/abs/10.1080/00207549608904937

Harding, J. A. and Popplewell, K. and Cook, D., 2003. Manufacturing system engineering moderator: An aid for multidiscipline project teams, International Journal of Production Research, 41 (9), pp. 1973-1986}, doi:10.1080/0020754031000077257, http://www.tandfonline.com/doi/abs/10.1080/0020754031000077257

Harding, J. A. and Popplewell, K. Lin, HK (2007) A Generation of Moderators from Single Product to Global e-Supply Chain, Chapter V. In Putnik, G and Cunha, M M eds, *Knowledge and Technology Management in Virtual Organizations: Issues, Trends, Opportunities and Solutions*, pp110-135, published by Idea Group Inc (IGI), ISBN: 1599041669, 9781599041667.

Harding, J. A., Das, B. P., Swarnkar, R., Palmer, C. and Young, R. I., 2011, Revised Report on design and implementation of Collaboration Moderator Services (SYNERGY Deliverable 3.3c), Project Report

Gamma, E., Helm, R., Johnson, R., Vlissides, J., 1995, Design patterns: Elements of reusable object-oriented design, Addison-Wesley Reading, MA

Khilwani, N., Harding, J. A. and Tiwari, M. K., 2011, Enterprise competence organization schema: publishing the published competences, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture June 2011 vol. 225 no. 6 921-942, doi: 10.1177/09544054JEM2097

Popplewell, K., 2008, "Supporting adaptive enterprise collaboration through semantic knowledge services". In *Enterprise interoperability III: New challenges and industrial approaches*, Edited by: Mertins, K, Ruggaber, R, Popplewell, K and Xu, X. 381–393. New York: Springer.

Violina Ratcheva, 2009, Integrating diverse knowledge through boundary spanning processes – The case of multidisciplinary project teams, International Journal of Project Management, Volume 27, Issue 3, April 2009, Pages 206-215, ISSN 0263-7863, 10.1016/j.ijproman.2008.02.008. http://www.sciencedirect.com/science/article/pii/S0263786308000379

Salazar, A., Hackney, R., Howells, J. 2003, The Strategic Impact of Internet Technology in Biotechnology and Pharmaceutical Firms: Insights from a Knowledge Management Perspective. Information Technology and Management, Springer Netherlands, Issn: 1385-951X, 4 (2), pp 289 - 301, Url: http://dx.doi.org/10.1023/A:1022910614411

Swarnkar, R., Choudhary, A., Harding, J., Das, B., Young, R., 2011. A framework for collaboration moderator services to support knowledge based collaboration, Journal of Intelligent Manufacturing, Springer Netherlands, Issn: 0956-5515, Url: http://dx.doi.org/10.1007/s10845-011-0528-2

SYNERGY, 2008, Synergy project, http://synergy-ist.eu, last retrieved 06/09/2012

SYNERGY Consortium, 2008, Conceptual Architecture (SYNERGY Deliverable 1.2). Project Report

Timothy J Norman, Alun Preece, Stuart Chalmers, Nicholas R Jennings, Michael Luck, Viet D Dang, Thuc D Nguyen, Vikas Deora, Jianhua Shao, W.Alex Gray, Nick J Fiddian, 2004, Agent-based formation of virtual organisations, Knowledge-Based Systems, Volume 17, Issues 2–4, May 2004, Pages 103-111, ISSN 0950-7051, 10.1016/j.knosys.2004.03.005. (http://www.sciencedirect.com/science/article/pii/S0950705104000061)

Walton, J. Whicker, L., Virtual Enterprise: Myth & Reality, 1996, CONTROL - COVENTRY - INSTITUTE OF OPERATIONS MANAGEMENT, VOL 22; NUMBER 8, pages 22-25

Appendix

TABLE 2: LIST OF ABBREVIATIONS FOR TERMS RELATED TO COLLABORATION MODERATOR

CM	Collaboration Moderator
CMKB	Collaboration Moderator Knowledgebase
CMS	Collaboration Moderator Service: CM provided as a service in the context of SYNERGY project
EM	Expert Module: a collection of knowledge units for a certain role of a certain user
IOI	Item of Interest: a knowledge unit
KAM	Knowledge Acquisition Module
RTM	Real Time Module
VO	Virtual Organisation
WM	Working Memory: a temporary knowledge repository which CM uses to store its knowledge about current processes it is monitoring